



SkillCourt Interactive Soccer Technology

CIS 4951 - Capstone II
Spring 2023



Introduction

SkillCourt Interactive Soccer Technology is an interactive tool for soccer players and teachers which consists of two major component types; a phone app and sensor pads. The phone app controls seven sensors at a time. The led indicators on the sensors tell the player which pads to hit and light up in order specific to the application's instructions.





Project Requirements (App and Web)

Mobile App

- Flutter (Channel stable, 3.7.10, on macOS 12.6.3 21G419 darwin-x64, locale en-US)
- Android toolchain - develop for Android devices (Android SDK version 33.0.0)
- Xcode - develop for iOS and macOS (Xcode 14.2)
- Chrome - develop for the web
- Android Studio (version 2021.2)
- VS Code (version 1.77.3)
- Connected device (2 available)
- HTTP Host Availability

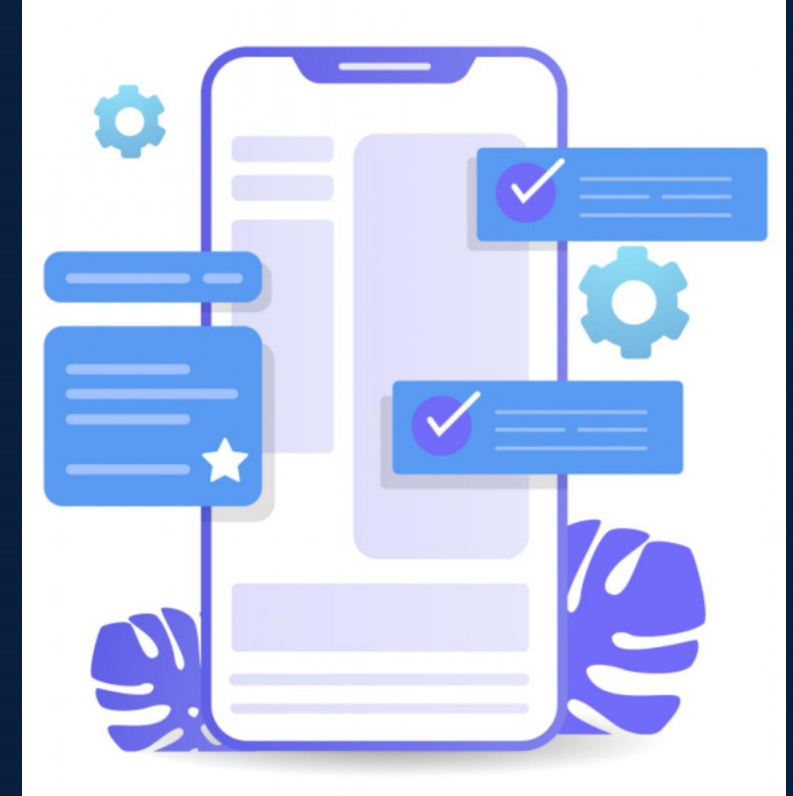
Website

- A GitHub Account
- Node $\geq$ v19.0.0
- Npm $\geq$ 9.3.1
- PNPM $\geq$ v6.21.0 or a V.M. with Linux installed
- Docker Desktop with PostgreSQL
- Vim (optional)
- VSCode (optional)



Project Goal (Mobile App)

- Update the phone application
- Fix bugs that were preventing developers from accessing and running the software on their phone emulators.
- Add more functionalities and features to the application
- Find a temporary solution to push new application updates to github.



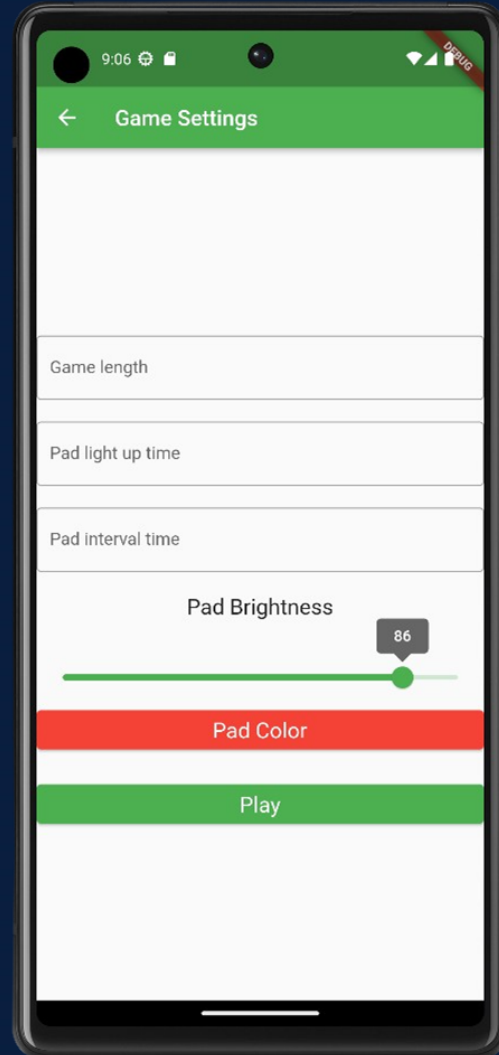


Contributions (Mobile App)

- A lot of work was done under the hood of the phone application.
- The updated applications include Dart, Android Tools, Gradle, and Flutter. Paths were edited, re-routed, and migrated.
- There were widgets within flutter that were broken and prevented the application from compiling. Those widgets had to be found and replaced by inserting new different references into the phone application's files.
- The github pre-commit file had to be edited for a temporary fix because it was preventing devs from pushing updates to github.
- Additions to the application UI, such as a brightness slider for the pad LEDs.



SkillCourt Interactive Soccer Tech App



```
ElevatedButton(  
  onPressed: () {  
    showDialog(  
      context: context,  
      builder: (BuildContext context) {  
        return AlertDialog(  
          title: Text('Pick a color!'),  
          content: SingleChildScrollView(  
            child: BlockPicker(  
              pickerColor: mycolor, //default color  
              onColorChanged: (Color color) {  
                //on color picked  
                setState(() {  
                  mycolor = color;  
                });  
              },  
            ), // BlockPicker  
          ), // SingleChildScrollView  
          actions: <Widget>[  
            ElevatedButton(  
              child: const Text('DONE'),  
              onPressed: () {  
                Navigator.of(context)  
                  .pop(); //dismiss the color picker  
              },  
            ), // ElevatedButton  
          ], // <Widget>[]  
        ); // AlertDialog  
      });  
    },  
  style: ElevatedButton.styleFrom(  
    backgroundColor: mycolor,  
    textStyle: const TextStyle(fontSize: 20)),  
  child: Text("Pad Color"),  
);
```

SkillCourt Interactive Soccer Tech App

PlatformButton

A button that will render a `RaisedButton` for android or a `CupertinoButton` for iOS.

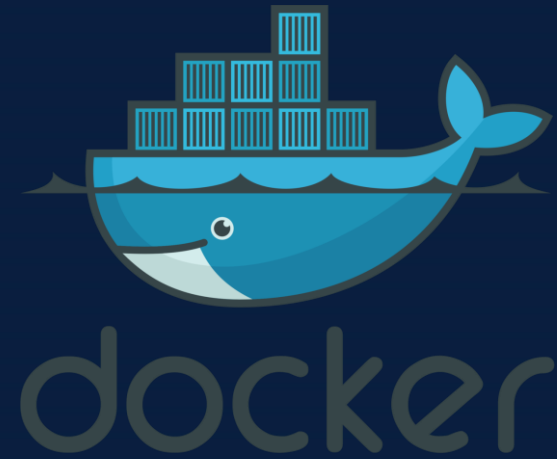
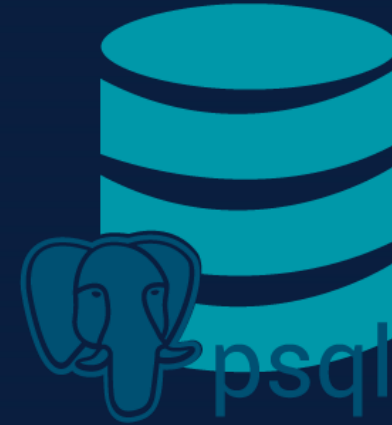
```
return PlatformButton(  
  onPressed: () => print('send'),  
  child: PlatformText('Send'),  
);
```

**Old Widget	change to	New Widget
FlatButton =>		TextButton
RaisedButton =>		ElevatedButton
OutlineButton =>		OutlinedButton**

- Various out of date Flutter button widgets were changed and adjusted

Project Goal (Website)

- Update documentation
- Update dependencies
- Log into database
- Run webapp locally
- Synchronize a bridge between the website and mobile application



Contributions (Website)

- The readme installation files from the SkillCourt's Github repository were out of date and needed to be updated by changing the docker compose installation instructions and methods.
- The Docker installation instructions were riddled with MySQL and MariaDB database references. Those references had to be changed to Postgres inside of the .env, and .yml files.
- Critical dependency versions had had to be updated, and the dependency tree was fixed.
- Prisma Schema issues were also fixed by updating the field data types.



Website Database Instances Replaced

```
services:
  webapp:
    build:
      context: .
      dockerfile: Dockerfile-devel
    networks:
      - webapp
    ports:
      - "4200:4200"
    volumes:
      - ./:/app
    environment:
      APP_HOST: ${APP_HOST}
      APP_JWT_SECRET: ${APP_JWT_SECRET}
      APP_ADMIN_EMAIL: ${APP_ADMIN_EMAIL}
      APP_ADMIN_PASSWORD: ${APP_ADMIN_PASSWORD}
      MYSQL_USER: ${MYSQL_USER}
      MYSQL_PASSWORD: ${MYSQL_PASSWORD}
      MYSQL_DATABASE: ${MYSQL_DATABASE}
      DATABASE_URL:
mysql: /${MYSQL_USER}:${MYSQL_PASSWORD}@database:3306/${MYSQL_DATABASE}
      SMTP_HOST: ${SMTP_HOST}
      SMTP_PORT: ${SMTP_PORT}
      SMTP_USER: ${SMTP_USER}
      SMTP_PASSWORD: ${SMTP_PASSWORD}
    depends_on:
      - database
    restart: unless-stopped

database:
  image: mariadb:10.7
  networks:
    - webapp
  ports:
    - 3306:3306
  environment:
    MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
    MYSQL_USER: ${MYSQL_USER}
```

(Before)

- Removed incorrect database references

```
version: '3'

services:
  webapp:
    build:
      context: .
      dockerfile: Dockerfile-devel
    networks:
      - webapp
    ports:
      - "4200:4200"
    volumes:
      - ./:/app
    environment:
      APP_HOST: ${APP_HOST}
      APP_JWT_SECRET: ${APP_JWT_SECRET}
      APP_ADMIN_EMAIL: ${APP_ADMIN_EMAIL}
      APP_ADMIN_PASSWORD: ${APP_ADMIN_PASSWORD}
      POSTGRES_USER: ${POSTGRES_USER}
      POSTGRES_PASSWORD: ${POSTGRES_PASSWORD}
      POSTGRES_DB: ${POSTGRES_DB}
      DATABASE_URL: postgres://${POSTGRES_USER}:${POSTGRES_PASSWORD}@database:5432/${POSTGRES_DB}
      SMTP_HOST: ${SMTP_HOST}
      SMTP_PORT: ${SMTP_PORT}
      SMTP_USER: ${SMTP_USER}
      SMTP_PASSWORD: ${SMTP_PASSWORD}
    depends_on:
      - database
    restart: unless-stopped
```

Mac Users The following commands must be run **inside** your virtual Linux operating system connected by `vagrant ssh`.

5. Launch the containers for the project by using `docker compose up`. Use `docker compose up -d` to launch the containers in a detached mode from the current shell. Add the option `--build` to rebuild the containers as needed.

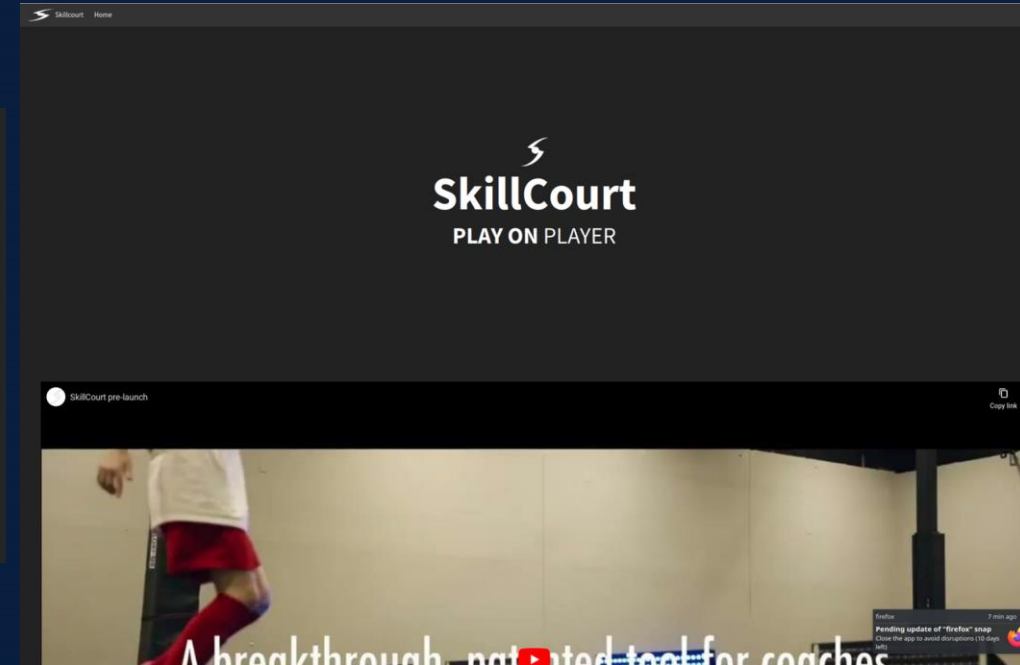
(After)

- Correct database references
- Updated docker compose instructions



Website Changes

```
2023-03-23T20:41:25.975345419Z ✓ Server application bundle generation complete
2023-03-23T20:41:26.139881493Z
2023-03-23T20:41:26.139192852Z 1 unchanged chunks
2023-03-23T20:41:26.139201789Z
2023-03-23T20:41:26.139204544Z Build at: 2023-03-23T20:41:25.959Z - Hash: 758c310cb5c3f39e - Time: 69710ms
2023-03-23T20:41:26.148844587Z
2023-03-23T20:41:26.148870459Z ./node_modules/.pnpm/@nestjs+common@9.2.0_mpbqhbv3qq77ujhdpg6d4hsy/node_modules/@nestjs/common/utils/load-package.ut
ll.js:16:35-55 - Warning: Critical dependency: the request of a dependency is an expression
2023-03-23T20:41:26.148873945Z
2023-03-23T20:41:26.148876231Z ./node_modules/.pnpm/@nestjs+core@9.2.0_46ufwkssjztvkmtvxmn5aildva/node_modules/@nestjs/core/helpers/load-adapter.js:1
6:35-59 - Warning: Critical dependency: the request of a dependency is an expression
2023-03-23T20:41:26.148878764Z
2023-03-23T20:41:26.148880949Z ./node_modules/.pnpm/@nestjs+core@9.2.0_46ufwkssjztvkmtvxmn5aildva/node_modules/@nestjs/core/helpers/optional-require.
js:10:35-55 - Warning: Critical dependency: the request of a dependency is an expression
2023-03-23T20:41:26.148883617Z
2023-03-23T20:41:26.148885686Z
2023-03-23T20:41:26.193297525Z
2023-03-23T20:41:26.193382138Z ./server/app.module.ts:7:23-54 - Error: Module not found: Error: Can't resolve '@nestjs/ng-universal' in '/app/server'
2023-03-23T20:41:26.193392259Z
2023-03-23T20:41:26.193394996Z Error: server/app.module.ts:5:40 - error TS2307: Cannot find module '@nestjs/ng-universal' or its corresponding type d
eclarations.
2023-03-23T20:41:26.193397640Z
2023-03-23T20:41:26.193399803Z 5 import { AngularUniversalModule } from '@nestjs/ng-universal';
2023-03-23T20:41:26.193402068Z
2023-03-23T20:41:26.193404356Z
2023-03-23T20:41:26.193406566Z
2023-03-23T20:41:26.193408664Z
2023-03-23T20:41:27.018293043Z ✓ Browser application bundle generation complete
2023-03-23T20:41:27.276471602Z this.debug is not a function
2023-03-23T20:41:27.871777550Z ✓ Server application bundle generation complete
2023-03-23T20:41:28.081201464Z ELIFECYCLE Command failed with exit code 1.
```



- Webapp container with outdated dependencies (on the left)
- once fixed the dependencies tree, we were able to log into webapp locally (on the right).



Website Changes

```
augusto@augusto-skillcourt:~/skillcourt-website$ sudo docker compose exec webapp pnpm prisma db push
.pnpm-store/v3/tmp/dlx-165 | +2 +
.pnpm-store/v3/tmp/dlx-165 | Progress: resolved 2, reused 2, downloaded 0, added 2, done
Environment variables loaded from .env
Prisma schema loaded from prisma/schema.prisma
Datasource "db": PostgreSQL database "agineek", schema "public" at "database:5432"
Error: db error: ERROR: foreign key constraint "clubs_ownerId_fkey" cannot be implemented
DETAIL: Key columns "ownerId" and "id" are of incompatible types: character varying and uuid.
0: sql_migration_connector::apply_migration::migration_step
  with step=AddForeignKey { foreign_key_id: ForeignKeyId(0) }
  at migration-engine/connectors/sql-migration-connector/src/apply_migration.rs:21
1: sql_migration_connector::apply_migration::apply_migration
  at migration-engine/connectors/sql-migration-connector/src/apply_migration.rs:10
2: migration_core::state::SchemaPush
  at migration-engine/core/src/state.rs:433

ERROR Command failed with exit code 1: prisma db push

pnpm: Command failed with exit code 1: prisma db push
at makeError (/usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:23351:17)
at handlePromise (/usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:23922:33)
at process.processTicksAndRejections (node:internal/process/task_queues:95:5)
at async Object.handler [as dlx] (/usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:219682:7)
at async /usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:227422:21
at async main (/usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:227389:34)
at async runPnpm (/usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:227644:5)
at async /usr/local/lib/node_modules/pnpm/dist/pnpm.cjs:227636:7
augusto@augusto-skillcourt:~/skillcourt-website$
```

agineek=# \dt

List of relations

Schema	Name	Type	Owner
public	clubs	table	agineek
public	games	table	agineek
public	groups	table	agineek
public	password_resets	table	agineek
public	players	table	agineek
public	stations	table	agineek
public	users	table	agineek

(7 rows)

agineek=#

- Wrong database data types would not create foreign keys (on the left).
- Fixing these instances allowed the devs to get into the database when prisma compiled (on the right).



Shortcomings (Web and App)

- We were not able to implement any functionality to the UI additions.
- Couldn't finish a stylized color picker, that chose one of five relative colors on a color spectrum.
- The entire team was unfamiliar with the languages, tools, and project documentation; thus, more time was spent updating the project to a working state than expanding the project.
- Focus was also diverted to reworking implementations from previous teams that were deemed unnecessary.

