Requirement Analysis

Created	@October 21, 2021 6:33 PM
Last Edited Time	@February 9, 2022 1:17 AM
Type	Technical Spec
Status	
Created By	
Last Edited By	
Stakeholders	

BLG 411E - Software Engineering

PROJECT PLAN

GROUP - 8

ILC

Furkan Hayta -

Ahmet Polat -

Utku Sabri Kaya -

Buğra Aydın -

Table of Contents

1. Introduction

In this documentation, a Requirements Analysis of the project is done in five parts. In the first three sections, the organization of the document, the features within the scope of the project are indicated with the necessary prerequisites, subsequent conditions, and use case scenarios. Among the Use cases, User Types, User Scenarios, Use Case Diagrams are detailed. In the fourth section, user interface models are modelled with mock-ups of the pages that are mapped to the user stories. In the last part, Flow diagrams are drawn. Flowcharts of each use case, E / R diagrams and general data flow of the system is shown. Also, important data considerations are mentioned.

2. System Requirements

2.1 The system returns license compatibility according to selected kinds of dependencies.

2.2 Repos can be accessed by the only project developer.

2.3 After an ILC search, the repo is automatically stored.

2.4 If entered project's license is **other** in GitHub, then the URL of the description about the license is returned.

3. Use Cases

3.1 User types

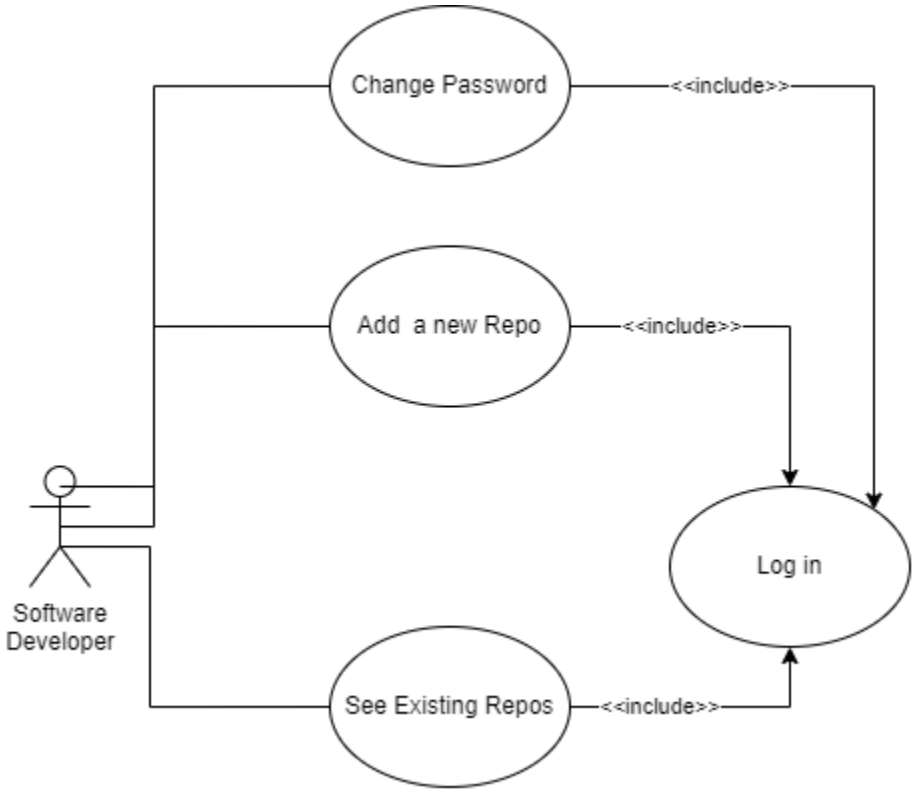
Software Developer → Can create a profile and add repositories to check license incompatibilities.

3.2 User scenarios

Madalyn needs to check her new social media project's license incompatibilities. She registers on the ILC website. She enters the GitHub URL of her project repository. In the license tree, she realizes that one of the dependent libraries restricts commercial use. She decides to write the functions imported from the library on her own.

Şadi adds a new dependency to his computer vision project. To be sure that the license compliance is still valid, he enters the ILC profile and selects the corresponding repository from his stored projects. Then, he sees the changes didn't affect license compatibility.

3.3 Use case diagram



3.4 Use cases

Use Case Name: Add a new Repo

Participating Actor: Software Developer

Pre-Condition: Software Developer must be log-in.

GitHub Repo must be public.

The remaining daily request limit must be greater than 0.

Post-Condition: None

Actor Actions	System Actions
1. Software developer enters the Github URL.	
2. Software developer selects include development dependencies or not.	3. System gets all dependencies.
	4. System checks license inconsistencies.
	5. System draws license graphs and lists all inconsistencies.

Alternative Flow of Events:

Step 3: Dependencies cannot be got. Display an error message.

Step 4: Unknown license type. Display an error message.

Exceptional Flow of Events:

The user leaves the page without waiting until the process is done. Stop the action.

Use Case Name: See Existing Repos

Participating Actor: Software Developer

Pre-Condition: Software Developer must be log-in.

The software developer must have at least 1 repo registered.

Post-Condition: None

Actor Actions	System Actions
1. Software developer clicks the My Repos.	2. System shows all repos of the user.
3. Software developer selects one repository.	4. System shows the details of the selected Repository.

Alternative Flow of Events:

Step 4: Repository details cannot be shown. Display an error message.

Exceptional Flow of Events: None

Use Case Name: Change Password

Participating Actor: Software Developer

Pre-Condition: Software Developer must be log-in.

Post-Condition: None

Actor Actions	System Actions
1. Software developer enters the current password and new password.	2. System changes the password as the given new password.

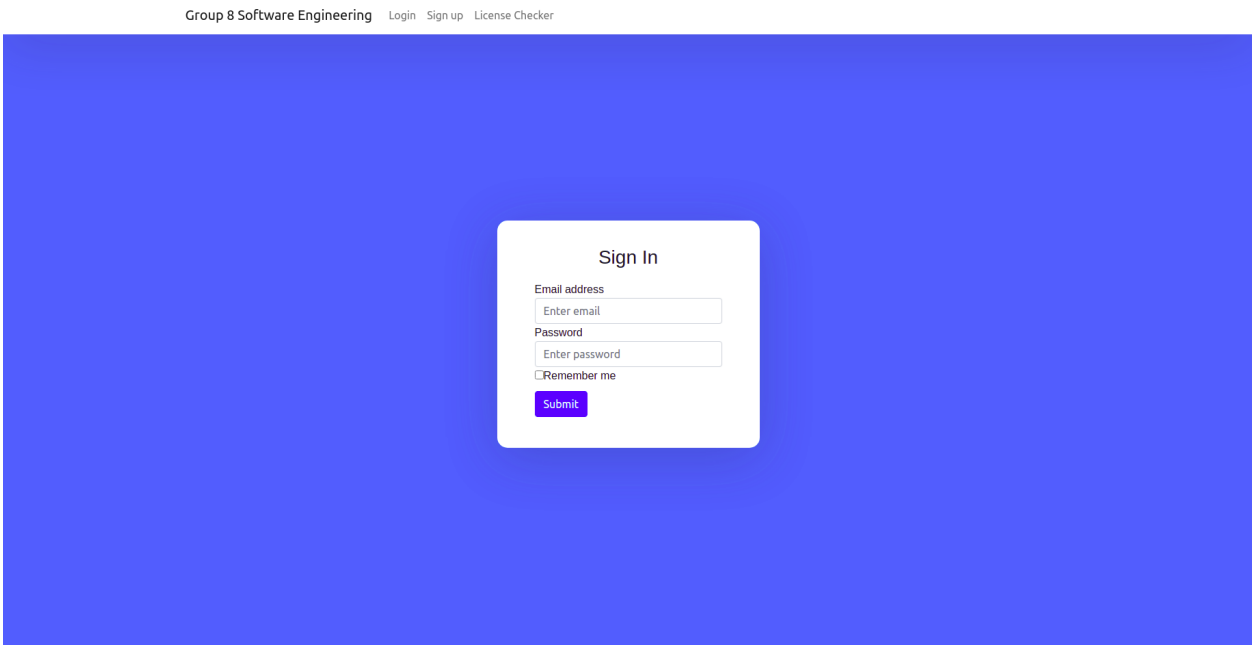
Alternative Flow of Events:

Step 2: New password violates the password rules. Display an error message.

Step 2: Current password is wrong. Display an error message.

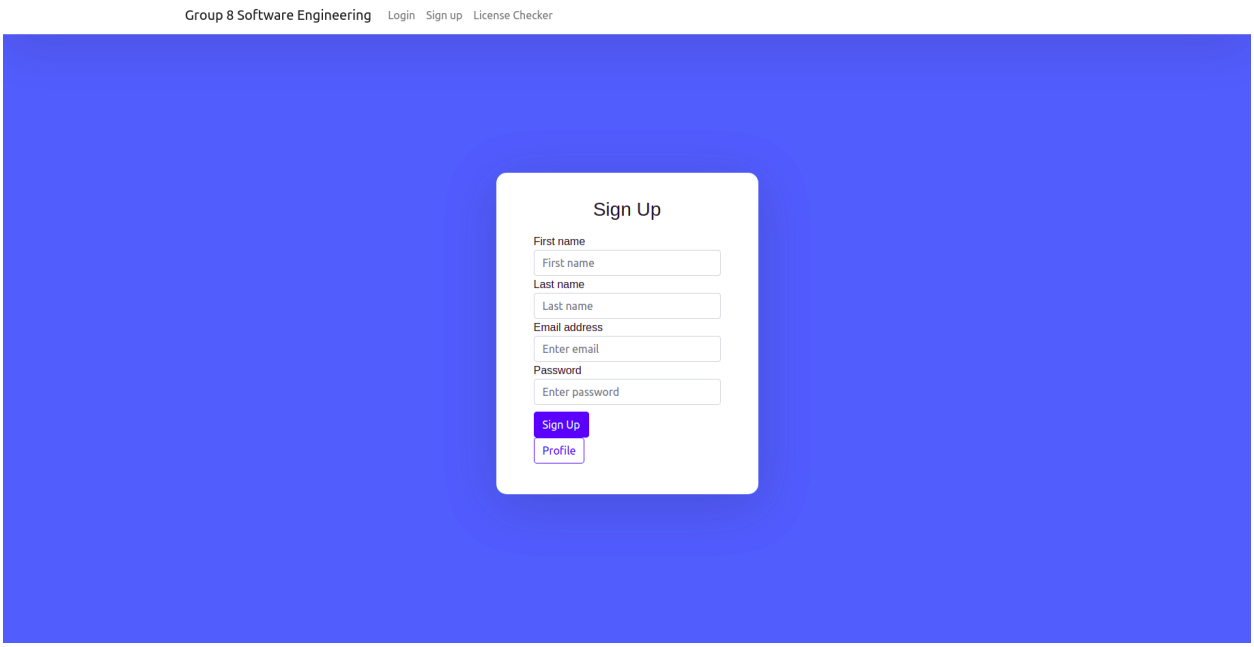
Exceptional Flow of Events: None

4. User Interface Model



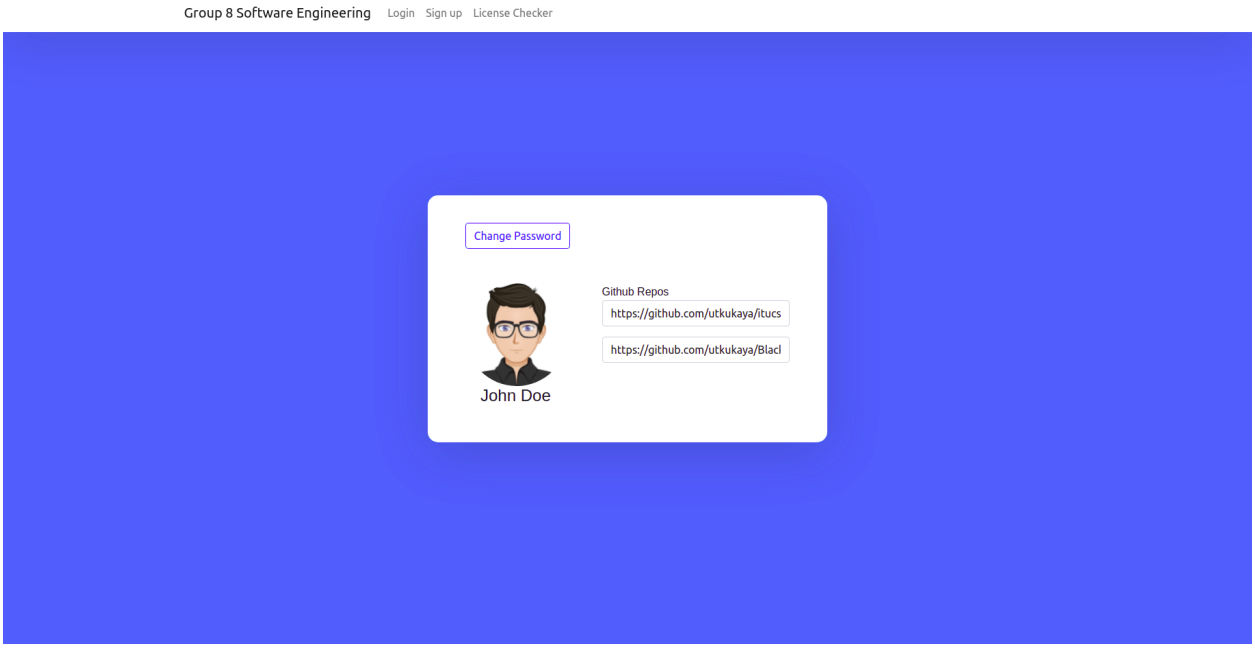
Login Screen

Users will be login by using this screen.



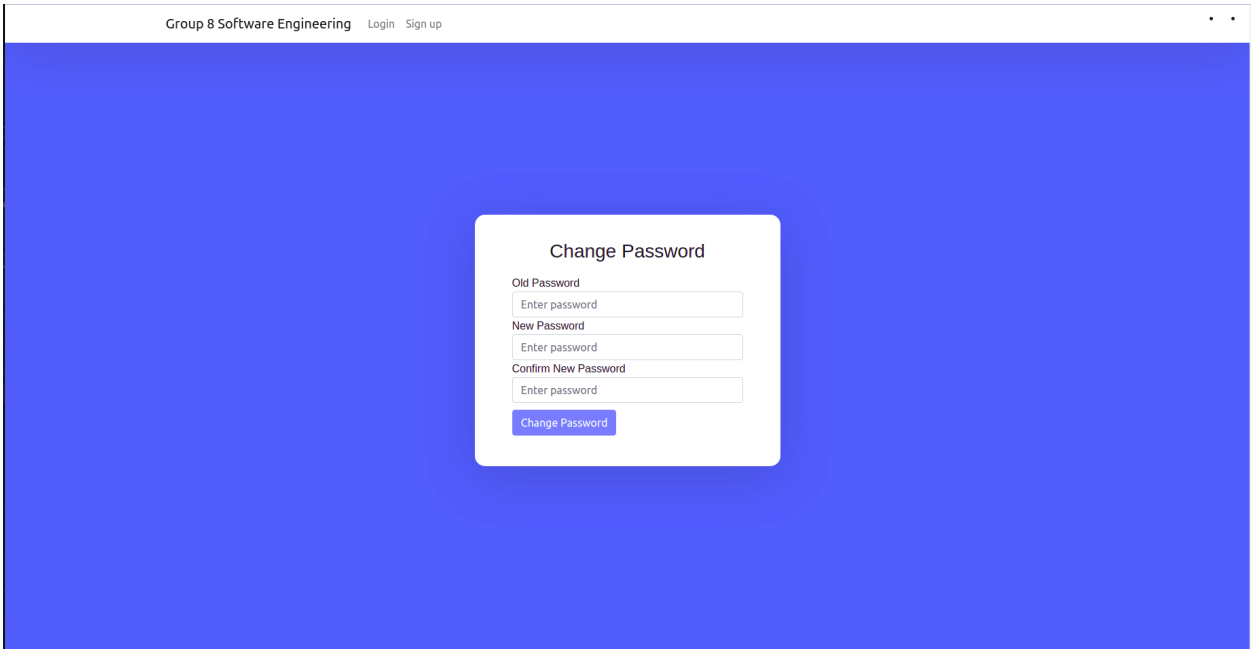
Sign up Screen

Users can register to our web app.



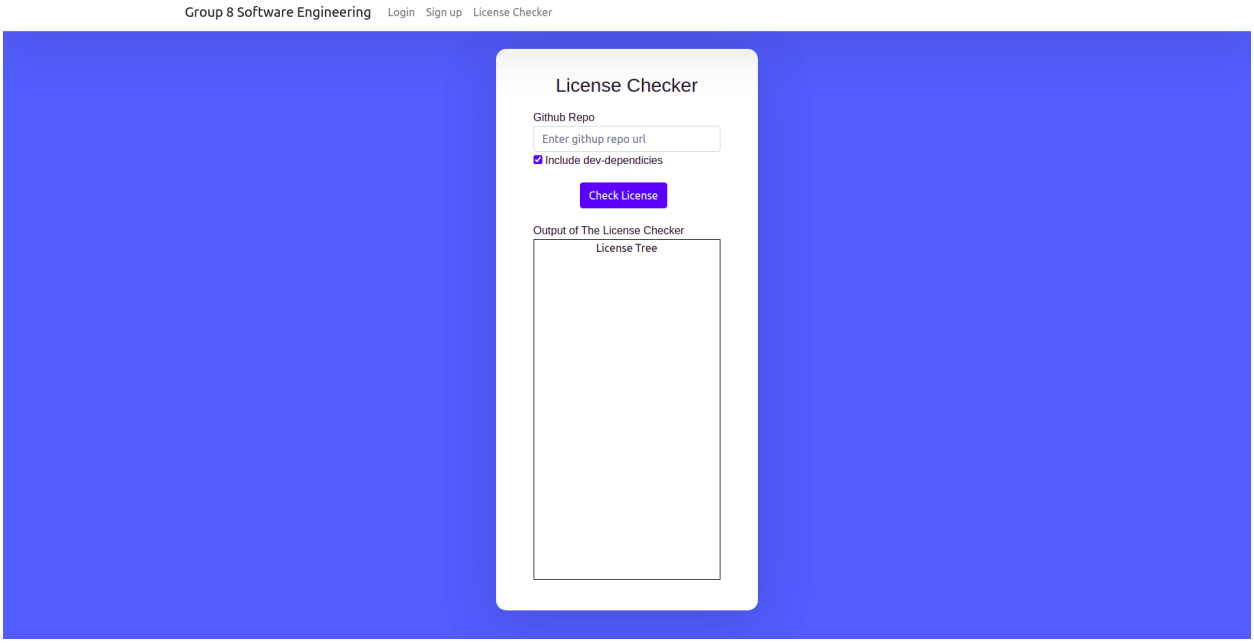
Profile Screen

The profile screen is not completed. This page will be enhanced and new features will be able to be added.



Change Password Screen

Users can change their passwords if they want.

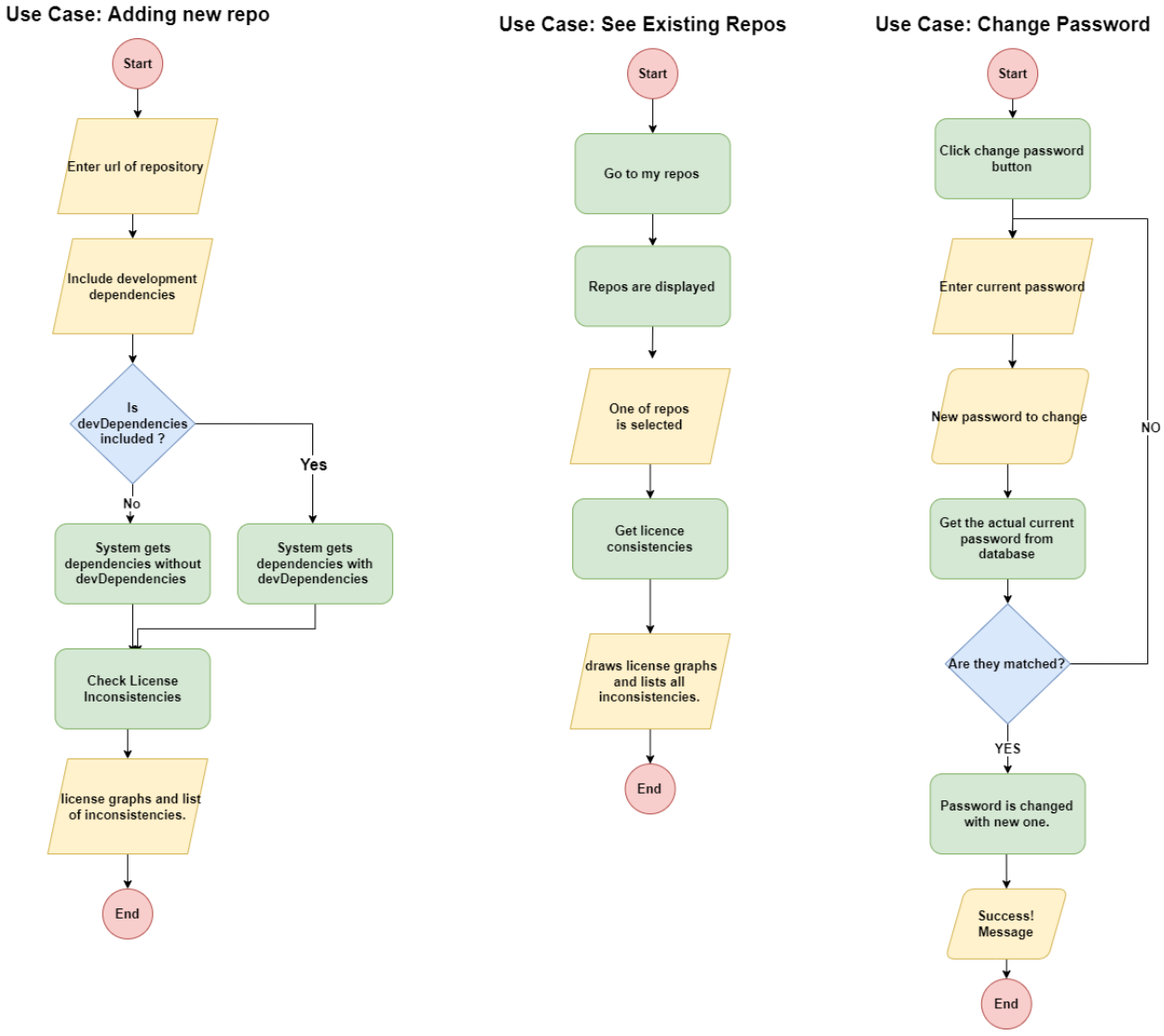


License Checker Screen

Users can be chosen that the project includes dev-dependencies. According to this option, the project will be checked.

5. Flow Diagrams

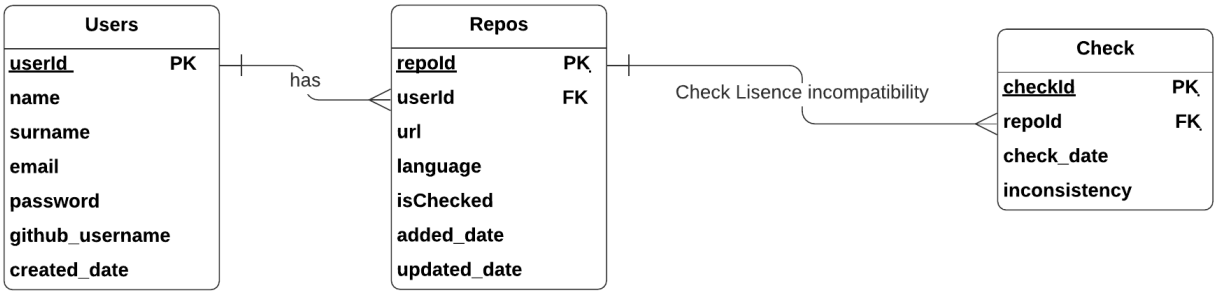
5.1 Flow Diagrams



5.2 General data model

Entity Relationship

Diagram of Database



There are 3 main tables in the database. Users have repos. This relationship is one to many.

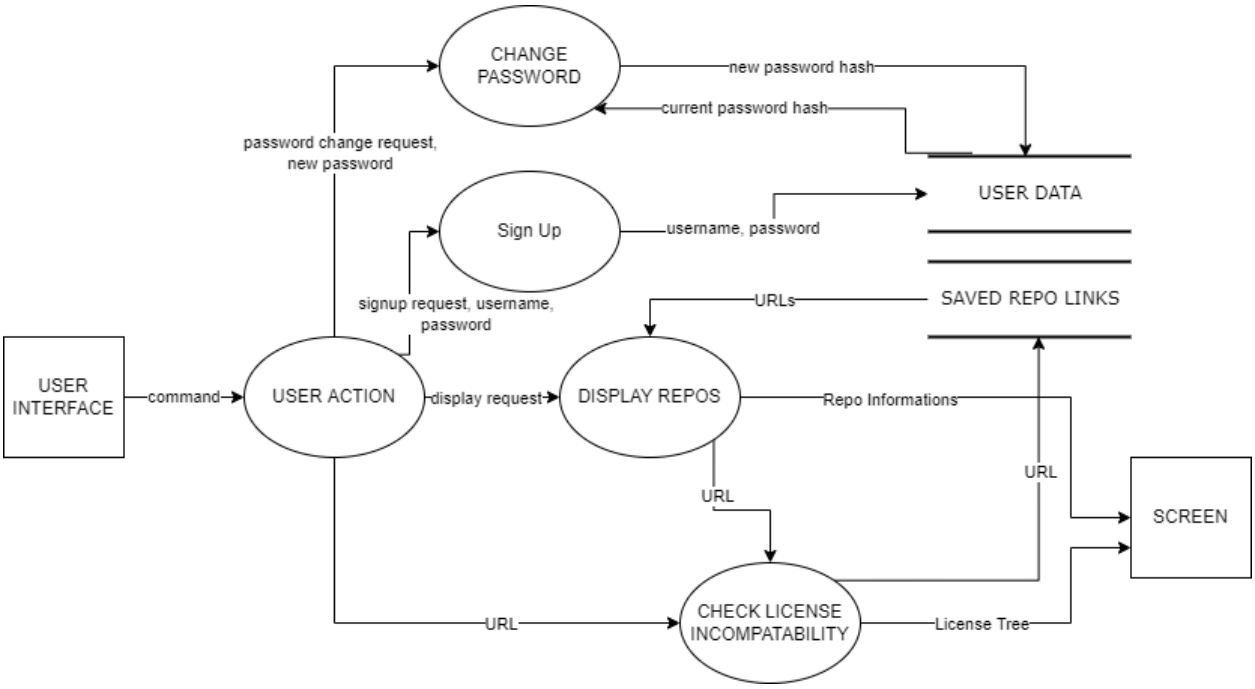
Users will be able to check the license of the repos they have uploaded. We'll keep this with the Check table. This relationship is provided as one to many.

5.3 Important data considerations

We need to set up the Backend API and perform data communication with the Frontend. We will perform this data communication in JSON data format. Data from forms and user directives will be modeled and transmitted in JSON format in accordance with a certain standard.

We also get the library licenses we get from the API for License Check in JSON format. We model the library license data we received and check the license inconsistency of the repo.

5.4 Data flow



Users can do four different fundamental operations in ILC. Firstly, they can sign up for the web app. Users can go sign-up page and send the sign-up form in Section 4. The given information is saved in the Users database in the E/R diagram. Secondly, users can change their passwords at any time. The Hash of the password is updated accordingly in the users' table. Also, Users can check their GitHub repo's license incompatibility giving the URL of the repo. An incompatibility graph is calculated and displayed to the user. Finally, Users can display all of their repos. In the Repos table, necessary data (URL) is getting and displayed to the user.

x