Student Graduation Predictor
Full Stack Application Report

Tagemouati Yassir —- Hassan Aya

June 12, 2025

1 System Overview

The application is a full-stack solution for predicting student graduation outcomes with:

- FastAPI backend with machine learning model
- React frontend with interactive dashboard
- Data visualization capabilities
- CSV upload and processing

2 Technical Stack

Layer	Technologies
Frontend	React, Vite, CSS, React Router
Backend	FastAPI, Uvicorn, Python 3.10
Machine Learning	Scikit-learn, Pandas, Joblib
Visualization	Matplotlib, Seaborn, react-slick
Build Tools	npm, Git

3 Project Structure

3.1 Backend Directory

```
backend/
          .idea/
          .venv/
          individual_plots/
          model/
                student_gb_model.pkl
          uploaded_data/
          data_generator.py
          main.py
          model_graduation.py
          plot_generator.py
```

3.2 Frontend Directory

```
frontend/
          public/
          src/
                assets/
                App.css
                App.jsx
                Footer.jsx
                Header.jsx
                Home.jsx
                main.jsx
                Prediction.jsx
                Sidebar.jsx
                Statistics.jsx
                Upload.jsx
          .gitignore
          package.json
          vite.config.js
```

4 Backend API Endpoints

Endpoint	Method	Description
`/predict`	POST	Predict graduation from student data
`/upload-data`	POST	Accept CSV uploads
`/generate-plots`	GET	Generate visualization plots

5 Machine Learning Model

- **Algorithm**: Gradient Boosting Classifier
- **Features**:
  - Numeric: Mark, Validated/Repeated/Failed Semesters
  - Categorical: Gender, School, Specialty
  - Binary: Scholarship
- **Hyperparameters**:

```
   GradientBoostingClassifier(
       n_estimators=300,
       learning_rate=0.05,
       max_depth=5,
       min_samples_split=20,
       min_samples_leaf=10
   )

```

6 Frontend Components

6.1 Core Components

Component	Functionality
App.jsx	Root component with routing
Header.jsx	Theme toggle and Logo
Sidebar.jsx	Responsive navigation menu
Home.jsx	Loading page
Prediction.jsx	Graduation prediction form
Upload.jsx	CSV upload and plot generation
Statistics.jsx	Data visualization dashboard

6.2 Key Features

- Dark/light theme toggle
- Responsive design for all PC's screen sizes
- File upload with progress feedback
- Loading wait loop
- A contact us bar in the bottom

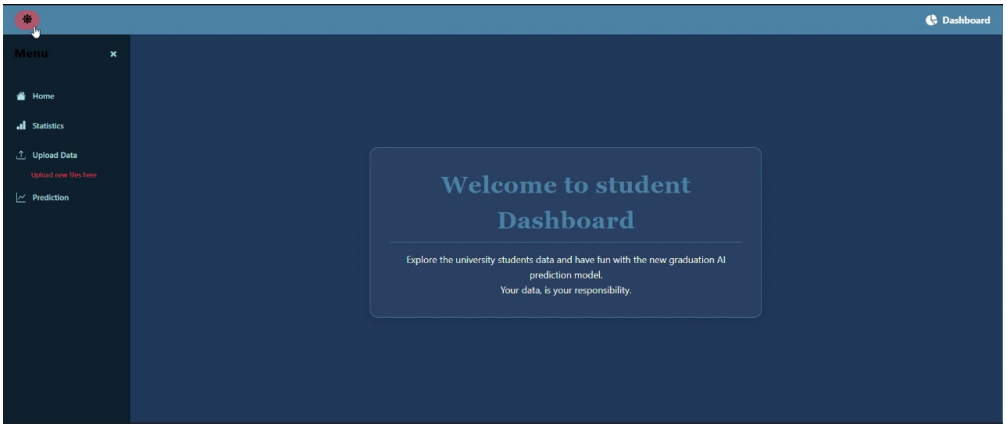

Figure 1: Home view

7 Data Flow

1. User uploads CSV via `Upload.jsx`
2. Frontend sends to `/upload-data` endpoint
3. Backend saves file and confirms receipt
4. User triggers plot generation

5. Backend processes data and generates visualizations
6. Frontend displays plots in carousel

8 Visualization Examples

The system generates four categories of plots:

- Academic Performance (Marks, Semesters)
- Student Demographics (Gender, Nationality)
- Institutional Performance (Schools, Specialties)
- Performance Trends (Over time)

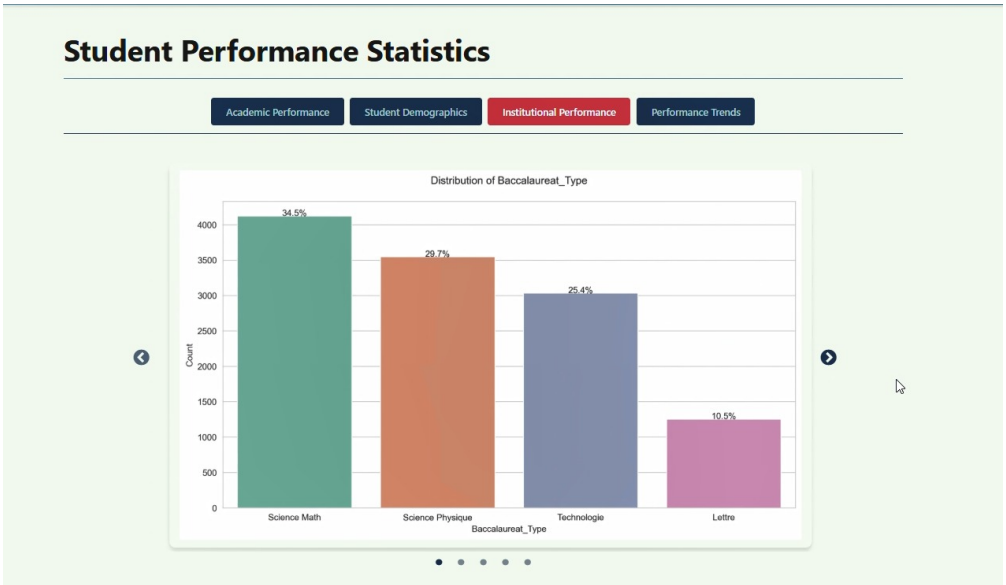

Figure 2: Sample visualization output

9 Deployment

```
# Backend
uvicorn main:app --host 0.0.0.0 --port 8000

# Frontend
npm run dev
```

10 Conclusion

This application provides:

- A prediction system that can tell wether the person is graduating in the future or not and also it is a small dashboard to read the statistics of the university students.

- I think if I want to improve the project I need to add a authentication for the user and also make the app more responsive for phone use but it is not harmful now because the app is dedicated to admins.