User Model (models/User.js):

```
const mongoose = require("mongoose");
const bcrypt = require("bcryptjs");

const userSchema = new mongoose.Schema({
  email: { type: String, required: true, unique: true },
  password: { type: String, required: true },
  balance: { type: Number, default: 1000 }
});

userSchema.pre("save", async function(next) {
  if (this.isModified("password")) {
    this.password = await bcrypt.hash(this.password, 10);
  }
  next();
});

module.exports = mongoose.model("User", userSchema);
```

Login Controller:

```
const jwt = require("jsonwebtoken");
const bcrypt = require("bcryptjs");
const User = require("../models/User");

exports.login = async (req, res) => {
  const { email, password } = req.body;
```

```
  const user = await User.findOne({ email });

  if (!user || !(await bcrypt.compare(password,
user.password))) {
    return res.status(401).json({ error: "Invalid credentials" });
  }

  const token = jwt.sign({ id: user._id }, "secretKey", {
expiresIn: "1h" });
  res.json({ token });
};
```

JWT Middleware (middleware/auth.js):

```
const jwt = require("jsonwebtoken");

module.exports = function(req, res, next) {
  const token = req.headers.authorization?.split(" ")[1];
  if (!token) return res.status(403).json({ error: "No token
provided" });

  try {
    const decoded = jwt.verify(token, "secretKey");
    req.userId = decoded.id;
    next();
  } catch (err) {
    res.status(401).json({ error: "Invalid token" });
  }
```

```
};
```